

数字电路

主讲:杨红权(13995601650)



第1章 数字逻辑基础

课程性质及学习方法:

作为后续课程(单片机原理及接口技术、DSP技术、计算机原理、计算机接口硬件等)的基础。

学习方法: (1). 搞清基本概念, 注意分析工作原理;
(2). 思维方式的训练;
(3). 独立完成作业, 按时交作业;
(4). 理论联系实际, 与实验相结合;
(5). 采用Protues等软件进行仿真相结合。



第1章 数字逻辑基础

数字技术的应用领域

消费电子



工业控制



电机控制
机床控制
生产过程自动化控制
楼宇电梯控制
.....

军用领域

第1章 数字逻辑基础

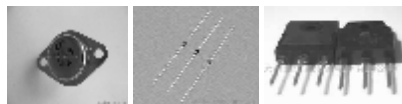
发展特点:以电子器件的发展为基础

电子管时代

1906年, 福雷斯特等发明了电子管; 电子管体积大、重量重、耗电大、寿命短。目前在一些大功率发射装置中使用。

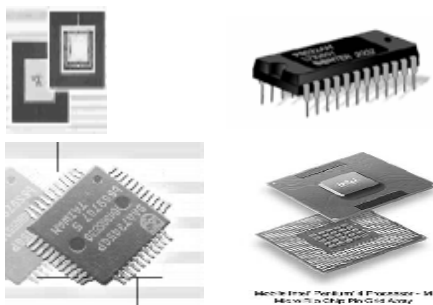


晶体管时代



第1章 数字逻辑基础

半导体集成电路



第1章 数字逻辑基础

数字电路的分类:

(1) 按集成度分类: 数字电路可分为小规模、中规模、大规模和超大规模(VLSI, 每片器件数目大于1万)数字集成电路。

(2) 按所用器件制作工艺的不同: 数字电路可分为双极型(TTL型)和单极型(MOS型)两类。

集成电路从应用的角度又可分为通用型和专用型两大类型。

定制器件和半定制器件。

EDA (Electronics Design Automation)技术

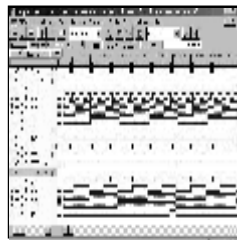
现代EDA技术实现硬件设计软件化。采用从上到下设计方法，电路设计、分析、仿真、修改全通过计算机完成。

EDA技术以计算机为基本工具、借助于软件设计平台，自动完成数字系统的仿真、逻辑综合等工作。最后下载到芯片，实现系统功能。

在计算机上利用软件平台进行设计



a. 仿真



b. 下载



c. 验证结果

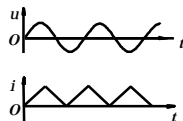
半定制器件。 → 可编程器件。

实验板

§ 1-1 数字信号的特点

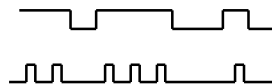
电子电路所处理的电信号可以分为两大类：

一类是在时间和数值上都是连续变化的信号，称为模拟信号，例如电流、电压信号等；



时间和数值均连续变化的信号，如正弦波、三角波等。

另一类是在时间和数值上都是离散的信号，为数字信号。传送和处理数字信号的电路，称为数字电路。



数字电路和模拟电路：工作信号，研究的对象，分析/设计方法以及所用的数学工具都有显著的不同。

§ 1-2 数制与码制

一、常用的进位计数制

通常把数的组成和由低位向高位进位的规则称为数制。

在数字系统中，常用的数制包括十进制数(decimal)，二进制数(binary)，八进制数(octal)和十六进制数(hexadecimal)。

十进制

数码为0~9，基数是10。计数规律为逢十进一，十进制数的权展开式：

$$\begin{array}{rcl}
 & \rightarrow & 5 \times 10^3 = 5000 \\
 & \rightarrow & 5 \times 10^2 = 500 \\
 & \rightarrow & 5 \times 10^1 = 50 \\
 & \rightarrow & 5 \times 10^0 = 5 \\
 & & + \\
 5 & 5 & 5 & 5 & & = 5555
 \end{array}$$

10³、10²、10¹、10⁰称为十进制的权。各数位的权是10的幂。

任意一个十进制数都可以表示为各个数位上的数码与其对应的权的乘积之和，称权展开式。

同样的数码在不同的数位上代表的数值不同。

$$\text{即: } (5555)_{10} = 5 \times 10^3 + 5 \times 10^2 + 5 \times 10^1 + 5 \times 10^0$$

二进制

数码为0、1，基数是2。计数规律为逢二进一，即：1+1=10，二进制数的权展开式：

$$\text{如：}(101.01)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = (5.25)_{10}$$

二进制数只有0和1两个数码，它的每一位都可以用电子元件来实现，且运算规则简单，相应的运算电路也容易实现。

运算规则

加法规则：0+0=0，0+1=1，1+0=1，1+1=10

乘法规则：0.0=0，0.1=0，1.0=0，1.1=1

八进制

数码为0~7，基数是8。计数规律为逢八进一，即：7+1=10。八进制数的权展开式：

$$\text{如：}(207.04)_8 = 2 \times 8^2 + 0 \times 8^1 + 7 \times 8^0 + 0 \times 8^{-1} + 4 \times 8^{-2} = (135.0625)_{10}$$

十六进制

数码为0~9、A~F，基数是16。计数规律为逢十六进一，即：F+1=10。十六进制数的权展开式：

$$\text{如：}(D8.A)_{16} = 13 \times 16^1 + 8 \times 16^0 + 10 \times 16^{-1} = 216.625)_{10}$$

二、不同数制间的转换

例：十进制数25转换成二进制的转换过程：

2	25	1/4	1/4	余 1	1/4	1/4	K ₀
2	12	1/4	1/4	余 0	1/4	1/4	K ₁
2	6	1/4	1/4	余 0	1/4	1/4	K ₂
2	3	1/4	1/4	余 1	1/4	1/4	K ₃
2	1	1/4	1/4	余 1	1/4	1/4	K ₄
0							

$$(25)_{10} = (11001)_2$$

二、八、十六进制间的转换

转换方法：(N)_B → (N)₁₀、(N)_H

将二进制数从小数点开始，分别向左、右按3位或4位分组，不足3位或4位的则需在最高位或最低位补0，最后将每组用对应的八进制数或十六进制数代替。

例如：八进制：	2	5	7	0	5	5	4
二进制：	010	101	111	000	101	101	100
十六进制：	A	F	1	6	C		

$$(257.0254)_{10} = (10101111.0001011011)_2 = (AF.16C)_H$$

§1-3 二进制编码

由于人们生活中习惯采用的是十进制，而数字电路便于采用的是二进制，这自然就提出了如何用二进制编码来表示十进制数的问题，即二-十进制编码的问题。

BCD - Binary Coded Decimal (二进制编码的十进制代码)

BCD码用四位二进制数表示0-9十个数码，四位二进制数最多可以表示16个字符，因此从16种表示中选十个来表示0-9十个字符，可以有多种情况。不同的表示法便形成了一种编码。这里主要介绍：

8421码 2421码

5421码 余3码

(N)_D = K₃W₃ + K₂W₂ + K₁W₁ + K₀W₀ W₃~W₀为各位的权重
所谓的8421码，就是指各位的权重是8、4、2、1。

8421 BCD码

8421BCD码是有权码，各位的权值分别为8，4，2，1。虽然8421BCD码的权值与四位自然二进制码的权值相同，但二者是两种不同的代码。8421BCD码只是取用了四位自然二进制代码的前10种组合。

8421BCD码在计算中应用非常广泛。

二进制数	8421码
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	
1011	
1100	
1101	
1110	
1111	

2421 BCD码

2421 BCD码的各位权值分别为2, 4, 2, 1, 2421码是有权码, 也是一种自补代码。

用BCD码表示十进制数时, 只要把十进制数的每一位数码, 分别用BCD码取代即可。反之, 若要知道BCD码代表的十进制数, 只要把BCD码以小数点为起点向左、向右每四位分一组, 再写出每一组代码代表的十进制数, 并保持原排序即可。

二进制数	2421码
0000	0
0001	1
0010	2
0011	3
0100	4
0101	
0110	
0111	
1000	
1001	
1010	
1011	5
1100	6
1101	7
1110	8
1111	9

5421 BCD码

5421BCD码是有权码, 各位的权值分别为5, 4, 2, 1。

二进制数	5421码
0000	0
0001	1
0010	2
0011	3
0100	4
0101	
0110	
0111	
1000	5
1001	6
1010	7
1011	8
1100	9
1101	
1110	
1111	

余3码

余3码是8421 BCD码的每个码组加3 (0011)形成的。余3码也具有对9互补的特点, 即它也是一种9的自补码, 所以也常用于BCD码的运算电路中。

二进制数	余3码
0000	
0001	
0010	
0011	0
0100	1
0101	2
0110	3
0111	4
1000	5
1001	6
1010	7
1011	8
1100	9
1101	
1110	
1111	

可靠性编码:

格雷码

格雷码 (Gray码) 也称循环码, 其最基本的特性是任何相邻的两组代码中, 仅有一位数码不同, 因而又叫单位距离码。

典型的Gray码编码方案如表所示。这种代码具有的一个特点就是反射特性, 即按表中所示的对称轴为界, 除最高位互补反射外, 其余低位数沿对称轴镜像对称。

二进制数	Gray码
0000	0000
0001	0001
0010	0011
0011	0010
0100	0110
0101	0111
0110	0101
0111	0100
1000	1100
1001	1101
1010	1111
1011	1110
1100	1010
1101	1011
1110	1001
1111	1000

奇偶校验码

代码(或数据)在传输和处理过程中, 有时会出现代码中的某一位由0错变成1, 或1变成0。奇偶校验码是一种具有检验出这种错误的代码, 奇偶校验码由信息位和一位奇偶检验位两部分组成。

检验位仅有一位, 它可以放在信息位的前面, 也可以放在信息位的后面。它的编码方式有两种:
使得一组代码中信息位和检验位中“1”的个数之和为奇数, 称为奇检验;
使得一组代码中信息位和检验位中“1”的个数之和为偶数, 称为偶检验。

带奇偶校验的8421 BCD码

十进制数	8421BCD	奇检验	8421BCD	偶检验
0	0000	1	0000	0
1	0001	0	0001	1
2	0010	0	0010	1
3	0011	1	0011	0
4	0100	0	0100	1
5	0101	1	0101	0
6	0110	1	0110	0
7	0111	0	0111	1
8	1000	0	1000	1
9	1001	1	1001	0

信息位 校验位 信息位 校验位

字符代码---ASCII码

ASCII码采用七位二进制数编码,因此可以表示128个字符。从表中可见,数字0-9,相应用0110000~0111001来表示,B8通常用作奇偶检验位,但在机器中表示时,常使其为0,因此0-9的ASCII码为30H~39H,大写字母A-Z的ASCII码为41H~5AH等。

十进制	二进制	十进制	二进制	十进制	二进制	十进制	二进制
0	0000000	10	0001010	20	00010100	30	00011110
1	0000001	11	0001011	21	00010110	31	00011111
2	0000010	12	0001100	22	00011000	32	00100000
3	0000011	13	0001101	23	00011010	33	00100001
4	0000100	14	0001110	24	00011100	34	00100010
5	0000101	15	0001111	25	00100001	35	00100011
6	0000110	16	0010000	26	00100010	36	00100100
7	0000111	17	0010001	27	00100011	37	00100101
8	0001000	18	0010010	28	00100100	38	00100110
9	0001001	19	0010011	29	00100110	39	00100111

§1-4 基本逻辑运算

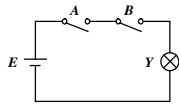
一、逻辑变量和逻辑函数

逻辑数学中,把相互对立的逻辑关系且仅有两个取值的变量称为逻辑变量。一般用“真”、“假”或者“1”、“0”来表示逻辑变量的取值。

虽然“1”与“0”叫逻辑值或逻辑常量,但并不代表具体的数值大小,而是表示两种相互对立的逻辑状态。如电灯的“亮”与“灭”、电动机的“旋转”与“停止”等对立状态。

二、基本逻辑运算

与逻辑关系:当决定某一事件的所有条件都具备时,事件才能发生。这种决定事件的因果关系称为“与逻辑关系”。



与逻辑关系表

A	B	Y
断	断	灭
断	通	灭
通	断	灭
通	通	亮

与逻辑真值表

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

其逻辑表达式(也叫逻辑函数式)为: 表 1.2 与逻辑真值表

$$Y=A \cdot B$$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

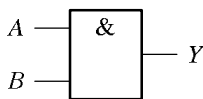
读作“Y等于A乘B”。在不致混淆的情况下,可以把符号“·”省掉。在有些文献中,也采用∩、∧、&等符号来表示逻辑乘。

由表2-1的真值表可知,逻辑乘的基本运算规则为:

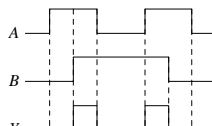
$$\begin{aligned} 0 \cdot 0 &= 0 & 0 \cdot 1 &= 0 & 1 \cdot 0 &= 0 & 1 \cdot 1 &= 1 \\ 0 \cdot A &= 0 & 1 \cdot A &= A & A \cdot A &= A \end{aligned}$$

与逻辑的运算规律为:输入有0得0,全1得1。与逻辑的逻辑符号如图所示。

与逻辑的波形图如图所示。该图直观地描述了任意时刻输入与输出之间的对应关系及变化的情况。

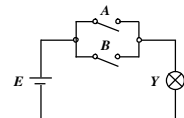


与逻辑符号图



与逻辑波形图

或逻辑关系:当决定事物结果的几个条件中,只要有一个或一个以上条件得到满足,结果就会发生,这种因果关系称为“或逻辑关系”。



或逻辑关系表

A	B	Y
断	断	灭
断	通	亮
通	断	亮
通	通	亮

或逻辑真值表

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

或运算也称“逻辑加”。或运算的逻辑表达式为：

$$Y = A + B$$

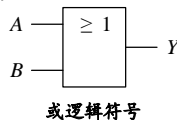
读作“F等于A加B”。有些文献也采用U、V等符号来表示逻辑加。

由真值表可知，逻辑加的运算规则为：

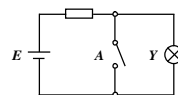
$$0+0=0 \quad 0+1=1 \quad 1+0=1 \quad 1+1=1$$

$$0+A=A \quad 1+A=1 \quad A+A=A$$

或逻辑运算的规律为：有1得1，全0得0。或逻辑的逻辑符号如图所示。



非逻辑关系：在事件中，结果总是和条件呈相反状态，这种逻辑关系称为“非逻辑关系”。



非逻辑电路图

非逻辑关系表

A	Y
断	亮
通	灭

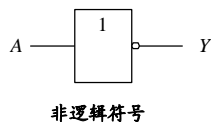
非逻辑真值表

A	Y
0	1
1	0

非运算也称“反运算”。非运算的逻辑表达式为：

$$Y = \overline{A}$$

运算的规律为：0变1，1变0，即“始终相反”。逻辑符号如图所示。



常见的复合逻辑关系

工程上常用的有：与非、或非、与或非、异或、同或。

与非运算 逻辑表达式为： $Y = \overline{AB}$

与非门的逻辑符号和真值表如下：

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

与非门的逻辑符号

真值表

有0得1，全1得0

常见的复合逻辑关系

或非运算 逻辑表达式为： $Y = \overline{A+B}$

或非门的逻辑符号和真值表如下：

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

或非门的逻辑符号

真值表

有1得0，全0得1

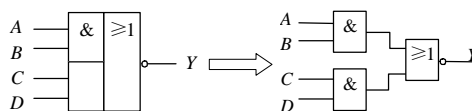
常见的复合逻辑关系

与或非运算 逻辑表达式为：

$$Y = \overline{AB+CD}$$

运算顺序：先进行与运算，再进行或非运算。

与或非门的逻辑符号如下：



与或非门的逻辑符号

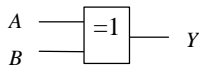
与或非门的等效电路

常见的复合逻辑关系

异或运算 逻辑表达式为: $Y = A \oplus B$

A, B不相同, Y为1; A, B相同时, Y为0

异或运算的逻辑符号和真值表如下:

		异或运算真值表	
A	B	Y	
0	0	0	
0	1	1	
1	0	1	
1	1	0	

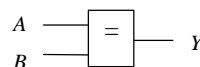
异或门的逻辑符号

常见的复合逻辑关系

同或运算 逻辑表达式为: $Y = A \odot B$

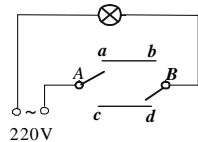
A, B相同时, Y为1; A, B不同时, Y为0

同或运算的逻辑符号和真值表如下:

		同或运算真值表	
A	B	Y	
0	0	1	
0	1	0	
1	0	0	
1	1	1	

同或门的逻辑符号

例: 下图是一个控制楼梯照明灯的电路。单刀双掷开关A装在楼下, B装在楼上, 这样在楼下开灯后, 可在楼上关灯; 同样也可在楼上开灯楼下关灯。试用一个逻辑函数来描述开关A、B与照明灯之间的关系。



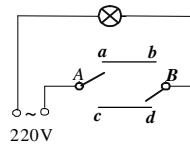
解:

设开关A、B为输入变量: 开关接上面为“1”, 开关接下面为“0”

设电灯L为输出变量, 灯亮L=1, 灯灭L=0。

(1) 列出A、B所有状态及对应输出L的状态, 即真值表。

(2) 根据真值表, 写出逻辑表达式。



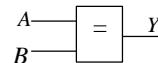
开关接上面为“1”,
开关接下面为“0”,
灯亮为“1”,
灯灭为“0”。

真值表为:

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	1

逻辑表达式为:

$Y = A \odot B$ 为同或运算



§ 1-4 逻辑代数的公式和规则

逻辑关系指的是事件产生的条件和结果之间的因果关系。在数字电路中往往是将事情的条件作为输入信号, 而结果用输出信号表示。条件和结果的两种对立状态分别用逻辑“1”和“0”表示。

逻辑代数又称布尔代数, 它是分析和设计现代数字逻辑电路不可缺少的数学工具。逻辑代数有一系列的定律、定理和规则, 用于对数学表达式进行处理, 以完成对逻辑电路的化简、变换、分析和设计。

一、基本公式

逻辑代数的基本公式

定律名称	逻辑与	逻辑或
1. 0-1律	$A \cdot 1 = A$ $A \cdot 0 = 0$	$A + 0 = A$ $A + 1 = 1$
2. 交换律	$A \cdot B = B \cdot A$	$A + B = B + A$
3. 结合律	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$	$A + (B + C) = (A + B) + C$
4. 分配律	$A \cdot (B + C) = A \cdot B + A \cdot C$	$A + (B \cdot C) = (A + B) \cdot (A + C)$
5. 互补律	$A \cdot \bar{A} = 0$	$A + \bar{A} = 1$
6. 重叠律	$A \cdot A = A$	$A + A = A$

逻辑代数的基本公式（续）

定律名称	逻辑与	逻辑或
7. 反演率 (摩根定律)	$\overline{AB} = \overline{A} + \overline{B}$	$\overline{A+B} = \overline{A} \cdot \overline{B}$
8. 吸收律	$A \cdot (A+B) = A$ $(A+B)(A+\overline{B}) = A$ $A(\overline{A}+B) = AB$	$A+A\overline{B} = A$ $AB+A\overline{B} = A$ $A+\overline{A}B = A+B$

二、逻辑代数的基本规则

1. 代入规则

在任何一个逻辑等式中，如果将等式两边的某一变量都用一个函数代替，则等式依然成立。这个规则称为代入规则。

例：已知等式 $\overline{AB} = \overline{A} + \overline{B}$

若用 $Y = B \cdot C$ 代替等式中的 B ，即：

$$\overline{A(BC)} = \overline{A} + \overline{B+C}$$

2. 反演规则

若求一个逻辑函数 Y 的反函数时，只要将函数中所有“·”换成“+”，“+”换成“·”；“0”换成“1”，“1”换成“0”；原变量换成反变量，反变量换成原变量；则所得到的逻辑函数式就是逻辑函数 Y 的反函数。

例：证明反演律 $\overline{A+B} = \overline{A} \cdot \overline{B}$

证：列出的真值表。

A	B	$\overline{A+B}$	$\overline{A} \cdot \overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

得证。

例：求下列函数的反函数。

$$Y_1 = \overline{A+B} \cdot \overline{B+C} + \overline{D+E}$$

$$Y_2 = \overline{A+BC+D} \cdot \overline{E}$$

注意：a. 遵守“先括号、然后与、最后或”的运算优先顺序；
b. 多个变量上的非号应保持不变

解： $\overline{Y_1} = \overline{\overline{A+B} \cdot \overline{B+C} + \overline{D+E}}$

$$\overline{Y_2} = \overline{A \cdot (B+C) \cdot D + E}$$

3. 对偶规则

Y 是一个逻辑表达式，如果将 Y 中的“·”换成“+”，“+”换成“·”，“0”换成“1”，“1”换成“0”，所得新逻辑函数式 Y' 为 Y 的对偶函数。

对于两个函数，如果原函数相等，那么其对偶函数、反函数也相等。

§ 1-6 逻辑函数的代数化简法

因为与或表达式比较常见且容易同其他形式的表达式相互转换，所以化简时一般要求化为最简与或表达式。

逻辑函数化简的方法有代数法和卡诺图法。代数法它可是直接运用基本定律及规则化简逻辑函数。方法有并项法、吸收法、消去法和配项法。

一、逻辑函数的最简与-或表达式

在若干个逻辑关系相同的与-或表达式中，将其中包含的与项数最少，且每个与项中变量数最少的表达式称为最简与-或表达式。

$$\begin{aligned}
 L &= AC + \overline{C}D && \text{“与-或”表达式} \\
 &= \overline{A}C \times \overline{C}D && \text{“与非-与非”表达式} \\
 &= (A + \overline{C})(C + D) && \text{“或-与”表达式} \\
 &= \overline{(A + \overline{C}) + (C + D)} && \text{“或非-或非”表达式} \\
 &= \overline{AC} + \overline{CD} && \text{“与-或非”表达式}
 \end{aligned}$$

二、公式化简法

并项法

利用 $A + \overline{A} = 1$ 的公式，将两项合并为一项，消去一个变量。

例：化简下列逻辑函数：

$$\begin{aligned}
 Y_1 &= \overline{A}BC + \overline{A}BC \\
 Y_2 &= \overline{A}BC + AB + \overline{A}C
 \end{aligned}$$

解：

$$\begin{aligned}
 Y_1 &= \overline{A}BC + \overline{A}BC = \overline{A}C(\overline{B} + B) = \overline{A}C \\
 Y_2 &= \overline{A}BC + AB + \overline{A}C = A(\overline{B}C + B + \overline{C}) \\
 &= A(\overline{B}C + \overline{B}C) = A
 \end{aligned}$$

吸收法

利用 $A + AB = A$ 的公式，消去多余的乘积项。

例：化简下列逻辑函数：

$$\begin{aligned}
 L_1 &= \overline{A}B + \overline{A}BCD(E + F) \\
 L_2 &= \overline{A}B + \overline{A}BC(D + E)
 \end{aligned}$$

解：

$$\begin{aligned}
 L_1 &= \overline{A}B + \overline{A}BCD(E + F) = \overline{A}B \\
 L_2 &= \overline{A}B + \overline{A}BC(D + E) \\
 &= \overline{A}B[1 + C(D + E)] \\
 &= \overline{A}B
 \end{aligned}$$

消去法

利用 $A + \overline{A}B = A + B$ ，消去多余的因子。

例：化简下列逻辑函数：

$$L = AB + \overline{A}C + \overline{B}C$$

解：

$$\begin{aligned}
 L &= AB + \overline{A}C + \overline{B}C \\
 &= AB + (\overline{A} + \overline{B})C && \boxed{A + \overline{B} = \overline{A}B} \\
 &= AB + \overline{A}BC && \boxed{A + AB = A + B} \\
 &= AB + C
 \end{aligned}$$

配项法

利用 $A = A(B + \overline{B})$ ，增加必要的乘积项，然后再用公式进行化简。

$$A + \overline{A} = 1$$

例：化简下列逻辑函数：

$$L = AB + \overline{A}\overline{C} + B\overline{C}$$

解：

$$\begin{aligned}
 L &= AB + \overline{A}\overline{C} + \overline{B}\overline{C} \\
 &= AB + \overline{A}\overline{C} + (\overline{A} + \overline{B})\overline{B}\overline{C} \\
 &= \overline{A}B + \overline{A}\overline{C} + \overline{A}\overline{B}\overline{C} + \overline{B}\overline{C} \\
 &= (\overline{A}B + \overline{A}\overline{B}\overline{C}) + (\overline{A}\overline{C} + \overline{A}\overline{B}\overline{C}) \\
 &= \overline{A}B + \overline{A}\overline{C}
 \end{aligned}$$

实际解题时，往往需要综合运用上述几种方法进行化简，才能得到最简结果。

例：化简逻辑函数 $Y_1 = \overline{A}\overline{C}B + \overline{A}\overline{C} + B + BC$

解：

$$\begin{aligned}
 Y_1 &= \overline{A}\overline{C}B + \overline{A}\overline{C} + B + BC \\
 &= \overline{A}\overline{C}B + \overline{A}\overline{C}B + BC && \text{（摩根定律）} \\
 &= \overline{A}\overline{C} + BC && \text{（合并法）} \\
 &= \overline{A} + C + BC && \text{（吸收法）} \\
 &= \overline{A} + C
 \end{aligned}$$

例：已知逻辑函数表达式为

$$L = AB\bar{D} + \bar{A}\bar{B}\bar{D} + ABD + \bar{A}\bar{B}\bar{C}D + \bar{A}\bar{B}CD$$

要求：

- (1) 最简的与-或逻辑函数表达式，并画出相应的逻辑图；
- (2) 仅用与非门画出最简表达式的逻辑图。

解： $L = AB(\bar{D} + D) + \bar{A}\bar{B}\bar{D} + \bar{A}\bar{B}D(\bar{C} + C)$

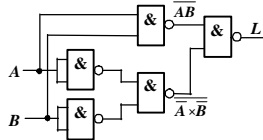
$$= AB + \bar{A}\bar{B}\bar{D} + \bar{A}\bar{B}D$$

$$= AB + \bar{A}\bar{B}(D + \bar{D})$$

$$= AB + \bar{A}\bar{B}$$

$$= \overline{AB + \bar{A}\bar{B}}$$

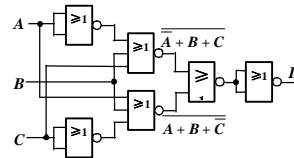
$$= \overline{AB \times \bar{A}\bar{B}}$$



例：试对逻辑函数表达式 $L = \overline{ABC} + \overline{ABC}$

进行变换，仅用或非门画出该表达式的逻辑图。

$$\begin{aligned} \text{解：} \quad L &= \overline{ABC} + \overline{ABC} = \overline{\overline{\overline{ABC} + \overline{ABC}}} \\ &= \overline{A + B + \bar{C} + A + B + C} \\ &= \overline{A + B + \bar{C} + A + B + C} \\ &= \overline{A + B + \bar{C} + A + B + C} \end{aligned}$$



三、卡诺图化简法

1. 逻辑函数的最小项

对n个输入变量的逻辑函数，共有 2^n 个最小项（乘积项）。

三变量的最小项编号

最小项	变量取值	最小项编号
\overline{ABC}	A B C	
\overline{ABC}	0 0 0	m_0
\overline{ABC}	0 0 1	m_1
\overline{ABC}	0 1 0	m_2
\overline{ABC}	0 1 1	m_3
\overline{ABC}	1 0 0	m_4
\overline{ABC}	1 0 1	m_5
\overline{ABC}	1 1 0	m_6
\overline{ABC}	1 1 1	m_7

乘积项中每个变量以原变量或反变量的形式仅出现一次。

2. 最小项的性质

三个变量的所有最小项的真值表

	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7
A B C	\overline{ABC}	\overline{ABC}	\overline{ABC}	\overline{ABC}	\overline{ABC}	\overline{ABC}	\overline{ABC}	\overline{ABC}
0 0 0	1	0	0	0	0	0	0	0
0 0 1	0	1	0	0	0	0	0	0
0 1 0	0	0	1	0	0	0	0	0
0 1 1	0	0	0	1	0	0	0	0
1 0 0	0	0	0	0	1	0	0	0
1 0 1	0	0	0	0	0	1	0	0
1 1 0	0	0	0	0	0	0	1	0
1 1 1	0	0	0	0	0	0	0	1

最小项性质：

- ①在输入变量的任何取值下有且仅有一个最小项的值为1。
- ②任意两个最小项的乘积为0。
- ③全体最小项之和为1。

最小项表达式：利用逻辑代数的基本公式，可以把任一逻辑函数化成一组最小项之和，称为最小项表达式。

例： $L(A, B, C) = AB + \bar{A}C = AB(C + \bar{C}) + \bar{A}C(B + \bar{B})$

$$= ABC + AB\bar{C} + \bar{A}BC + \bar{A}\bar{B}C$$

$$= m_1 + m_3 + m_6 + m_7 = \sum m(1, 3, 6, 7)$$

$$L(A, B, C) = (AB + \bar{A}B + C)\bar{A}B = (\bar{A}B + \bar{A}B + C) + \bar{A}B$$

$$= \bar{A}B \cdot \bar{A}B \cdot C + AB = (\bar{A} + B) \cdot (A + B) \cdot C + AB$$

$$= \bar{A}BC + \bar{A}\bar{B}C + AB = \bar{A}BC + \bar{A}\bar{B}C + AB(C + \bar{C})$$

$$= \bar{A}BC + \bar{A}\bar{B}C + ABC + AB\bar{C}$$

$$= m_3 + m_5 + m_6 + m_7 = \sum m(3, 5, 6, 7)$$

任一函数都可以化成唯一的最小项表达式

任何一个逻辑函数都可以表示成若干个最小项之和的形式。

例：写出三变量函数 $Y(A, B, C) = \overline{AB + \bar{A}B + C} + \bar{A}B$ 的最小项表达式。

解：利用摩根定律将函数变换为与或表达式，然后展开成最小项之和形式。

$$Y(A, B, C) = \overline{AB + \bar{A}B + C} + \bar{A}B$$

$$= \overline{AB + \bar{A}B} + \bar{A}B$$

$$= (\bar{A}B + \bar{A}B)\bar{C} + \bar{A}B(C + \bar{C})$$

$$= \bar{A}BC + \bar{A}\bar{B}C + \bar{A}B\bar{C} + \bar{A}B\bar{C}$$

$$= \bar{A}BC + \bar{A}\bar{B}C + \bar{A}B\bar{C}$$

$$= \sum m(2, 3, 4)$$

3. 用卡诺图表示逻辑函数

卡诺图的引出

卡诺图是指按相邻性原则排列的最小项的方格图。

卡诺图的特点

几何相邻对应着逻辑相邻

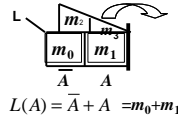
几何相邻——某一方格和其它方格具有共同的边

逻辑相邻——对于两个最小项，组成它们的变量中，只有一个不同，其余都相同。

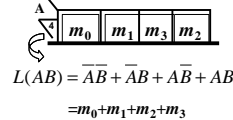
$$\begin{array}{ll} \text{如: } ABC, \overline{ABC} & \overline{ABCD}, \overline{ABCD} \\ C + \overline{C} = 1 & A + \overline{A} = 1 \end{array}$$

卡诺图的结构特点是按逻辑相邻进行排列，卡诺图的变量标注均采用循环码形式。

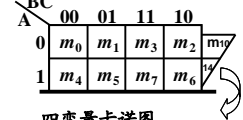
一变量卡诺图



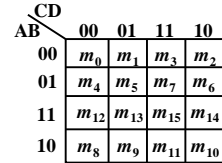
两变量卡诺图



三变量卡诺图



四变量卡诺图



由逻辑函数画卡诺图的方法:

a. 将逻辑函数化为最小项表达式; b. 填写卡诺图。

例: 用卡诺图表示逻辑函数 $L = \overline{A}B + \overline{A}BC + AC$

解: a. 将逻辑函数化为最小项表达式。

$$\begin{aligned} L &= \overline{A}B + \overline{A}BC + AC \\ &= \overline{A}B(C + \overline{C}) + \overline{A}BC + AC(B + \overline{B}) \\ &= \overline{A}BC + \overline{A}B\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + \overline{A}BC + \overline{A}B\overline{C} \\ &= \sum m(0, 2, 3, 4, 6) \end{aligned}$$

b. 填写卡诺图。

有最小项的地方用1表示，否则用0表示。

4. 用卡诺图化简逻辑函数

化简的依据:

$$\overline{ABCD} + \overline{ABCD} = \overline{ABD}$$

$$\overline{ABCD} + \overline{ABCD} = \overline{ABD}$$

$$\overline{ABD} + \overline{ABD} = \overline{AD}$$

$$\overline{ABD} + \overline{ABD} = \overline{AD}$$

$$\overline{AD} + \overline{AD} = D$$

方框格相邻时，总有互补变量出现，所以总能消去这一互补变量，使变量因子数减小。

例: 用卡诺图表示逻辑函数

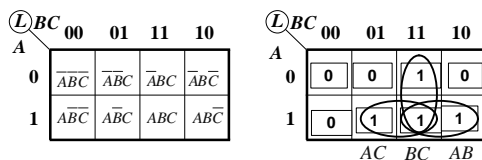
$$\begin{aligned} L(A, B, C) &= \overline{A}BC + \overline{A}BC + \overline{A}BC + \overline{A}BC \\ &= m_3 + m_5 + m_6 + m_7 = \sum (3, 5, 6, 7) \\ &= AC + BC + AB \end{aligned}$$

化简的步骤

a. 画出逻辑函数的卡诺图;

b. 合并最小项，即将相邻的为1的方格圈成一组;

c. 将所有包围圈对应的乘积项相加。



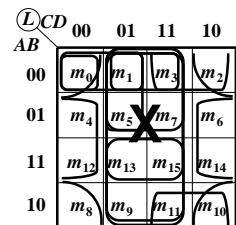
化简应遵循的原则:

a. 包围圈内的方格数一定是 2^n 个，且包围圈必须呈矩形;

b. 循环相邻特性包括上下底相邻，左右边相邻和四角相邻;

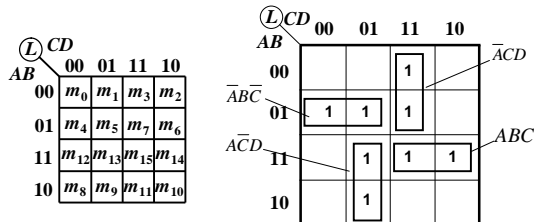
c. 同一方格可以被不同的包围圈重复包围多次，但新增的包围圈中一定要有原有包围圈未曾包围的方格;

d. 卡诺图越大越好，卡诺图越少越好。



例：化简 $L(A, B, C, D) = \sum m(3, 4, 5, 7, 9, 13, 14, 15)$ 。

解：画函数的卡诺图，化简过程如图所示。

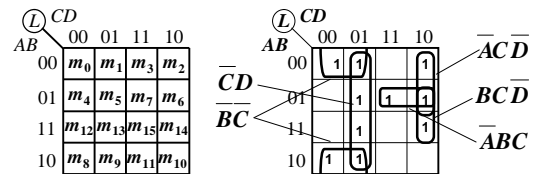


合并最小项得到的逻辑表达式为：

$$L = \overline{A}BC + \overline{A}CD + A\overline{C}D + ABC$$

例：用卡诺图化简逻辑函数

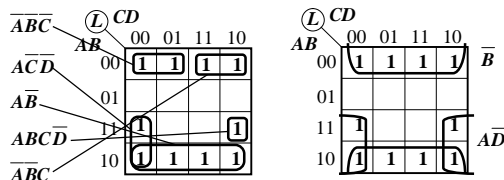
$$L(A, B, C, D) = \sum m(0, 1, 2, 5, 6, 7, 8, 9, 13, 14)$$



$$L = \overline{C}D + \overline{B}\overline{C} + \overline{A}BC + \overline{A}C\overline{D} + BCD$$

例：用卡诺图化简逻辑函数

$$L = \overline{A}B\overline{C} + A\overline{C}\overline{D} + A\overline{B} + ABC\overline{D} + \overline{A}\overline{B}C$$



$$L = \overline{B} + A\overline{D}$$

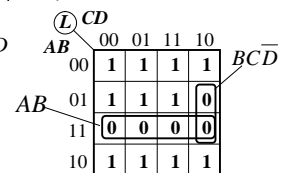
例：用卡诺图化简逻辑函数

$$L(A, B, C, D) = (A + \overline{B} + \overline{C} + D)(\overline{A} + \overline{B} + C + D)(\overline{A} + \overline{B} + C + \overline{D})(\overline{A} + \overline{B} + \overline{C} + D)(\overline{A} + \overline{B} + C + \overline{D})(\overline{A} + \overline{B} + \overline{C} + \overline{D})$$

解：a. 将逻辑函数化为最小项表达式。

$$\begin{aligned} \overline{L} &= \overline{A}B\overline{C}\overline{D} + A\overline{B}C\overline{D} + A\overline{B}C\overline{D} \\ &\quad + A\overline{B}C\overline{D} + A\overline{B}C\overline{D} \\ &= \sum m(6, 12, 13, 14, 15) \end{aligned}$$

b. 填写卡诺图。

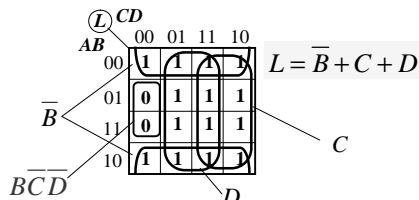


得逻辑表达式为：

$$Y = \overline{A}B + \overline{B}C\overline{D}$$

例：用卡诺图化简逻辑函数

$$L(A, B, C, D) = \sum m(0 \sim 3, 5 \sim 11, 13 \sim 15)$$



$$\overline{L} = \overline{B}C\overline{D} \quad L = \overline{B} + C + D$$

画包围圈时，可包围1，也可包围0

四、具有约束项的逻辑函数的化简

1. 约束项

逻辑变量之间的约束关系，如电机的正转、反转、停止；十字路口的红绿灯等。在同一时刻只能进行一种操作。

在实际的逻辑问题中，有些变量的取值是不允许、不可能、不应该出现的，这些取值对应的最小项称为约束项，有时又称为禁止项、无关项、任意项，在卡诺图或真值表中用 \times 或 Φ 来表示。

约束项的输出是任意的，可以认为是“1”，也可以认为是“0”。对于含有约束项的逻辑函数的化简，如果它对函数化简有利，则认为它是“1”；反之，则认为它是“0”。

逻辑函数中的约束项表示方法如下：如一个逻辑函数的约束项是：

$$\overline{ABC}, \overline{ABC}, \overline{ABC}, \overline{ABC}$$

则可以写成下列等式：

$$\overline{ABC} + \overline{ABC} + \overline{ABC} + \overline{ABC} = 0$$

2. 具有约束项的函数化简

如上述约束项可表示为：

$$\Sigma d(0, 2, 6, 7) = 0$$

具有约束项的化简步骤如下：

- 填入具有约束项的函数卡诺图；
- 画卡诺图合并（约束项“×”使结果简化看作“1”，否则为“0”）；
- 写出简化结果（消去不同，保留相同）。

例：已知 $Y = \overline{ACD} + \overline{ACD} + \overline{ABCD} + \overline{ABCD}$ ，约束条件为： $\overline{ABD} + CD = 0$ ，求最简的函数表达式。

解：（1）根据约束条件：

$$\overline{ABD} + CD = 0$$

配项展开，根据与或表达式和约束条件画卡诺图。

（3）画卡诺图，约束项可以视为“0”或者为“1”。

$$\Rightarrow Y = D + \overline{AB} + \overline{AC}$$

$$\overline{ABD} + CD = 0 \quad (\text{约束条件})$$

	CD	00	01	11	10
AB	00	1	1	X	
	01	1	X	X	1
	11		1	X	
	10		1	X	

例：已知 $Y(A, B, C, D) = \Sigma m(0, 2, 7, 8, 13, 15) + \Sigma d(1, 5, 6, 9, 10, 11, 12)$ 求最简的函数表达式。

解：（1）根据最小项表达式画卡诺图：

	CD	00	01	11	10
AB	00	m_0	m_1	m_3	m_2
	01	m_4	m_5	m_7	m_6
	11	m_{12}	m_{13}	m_{15}	m_{14}
	10	m_8	m_9	m_{11}	m_{10}

（2）画卡诺图，得逻辑函数表达式：

$$Y = BD + \overline{BD}$$

例：十字路口的交通信号灯，红、绿、黄灯分别用A、B、C表示；灯亮用1来表示，灯灭用0来表示。车辆通行状态用Y来表示，停车时Y为0，通车时Y为1。用卡诺图化简此逻辑函数。

解：灯全灭时，允许车辆感到安全时可以通行。

根据题目要求列出真值表，画卡诺图。

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	×
1	0	0	0
1	0	1	×
1	1	0	×
1	1	1	×

	BC	00	01	11	10
A	0	1	0	×	1
	1	0	×	×	×

画卡诺图合并最小项得：

$$Y = \overline{AC}$$

无关项：

真值表内对应于某些变量组合，函数值可以是任意的。或者说，这些变量组合根本不会出现，则这些变量组合对应的最小项称为无关项，也称任意项。所谓任意项就是，其取值是任意的，可取“1”，也可取“0”。

在卡诺图中的处理方法：

- 填卡诺图时，在对应的方格内填任意符号“×”。
- 化简时根据需要可将“×”视为“1”，也可视为“0”。

$$L(A, B, C, D) = m(5, 6, 7, 8, 9) + d(10, 11, 12, 13, 14, 15)$$

例：用卡诺图化简逻辑函数

$$L(A, B, C, D) = m(5, 6, 7, 8, 9) + d(10, 11, 12, 13, 14, 15)$$

1、画出逻辑函数的卡诺图

化简时可根据需要视为“1”也可视为“0”，使函数化到最简。

	CD	00	01	11	10
AB	00	m_0	m_1	m_3	m_2
	01	m_4	m_5	m_7	m_6
	11	m_{12}	m_{13}	m_{15}	m_{14}
	10	m_8	m_9	m_{11}	m_{10}

	CD	00	01	11	10
AB	00	0	0	0	0
	01	0	1	1	1
	11	×	×	×	×
	10	1	1	×	×

$$L = A + BC + BD$$

例：设计一个逻辑电路，能够判断1位十进制数是奇数还是偶数，是奇数，电路输出为1，偶数，电路输出为0。

$$L = \sum m(1, 3, 5, 7, 9) + \sum d(10, 11, 12, 13, 14, 15)$$

十进制数	A	B	C	D	L
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	1
10	1	0	1	0	×
11	1	0	1	1	×
12	1	1	0	0	×
13	1	1	0	1	×
14	1	1	1	0	×
15	1	1	1	1	×

无关项

$$L = D$$

逻辑函数及其表示方法

1. 逻辑函数

一般函数，当A, B, C, ...的取值确定之后，Y的值也就确定了。Y称为A, B, C, ...的函数。一般表达式可以写为：

$$Y = L(A, B, C, \dots)$$

与、或、非是三种基本的逻辑运算，即三种基本的逻辑函数。

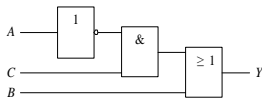
2. 逻辑函数的表示方法及转换

逻辑函数可以用逻辑真值表、逻辑表达式、逻辑图、波形图、卡诺图等方法来表示。

例：已知函数的逻辑表达式 $Y = B + \bar{A}C$ ，要求列出相应的真值表；已知输入波形，画出输出波形；画出逻辑图。

解：

(1) 根据逻辑表达式，画出逻辑图如图所示。

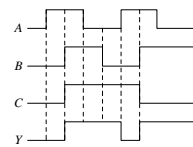


例：已知函数的逻辑表达式 $Y = B + \bar{A}C$ ，要求列出相应的真值表；已知输入波形，画出输出波形；画出逻辑图。

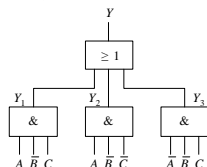
解：

(2) 将A, B, C的所有组合代入逻辑表达式中进行计算，得真值表如表所示。并绘出波形图如下。

A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



例：已知逻辑图如图所示，写出函数Y的逻辑表达式。



解：据逻辑图逐级写出输出端函数表达式如下：

$$Y_1 = A \bar{B} C \quad Y_2 = A \bar{B} \bar{C} \quad Y_3 = \bar{A} B \bar{C}$$

最后得到函数Y的表达式为：

$$Y = A \bar{B} C + A \bar{B} \bar{C} + \bar{A} B \bar{C}$$

本章小结

逻辑代数是按一定逻辑规律进行运算的、反映逻辑变量运算规律的一门数学，它是分析和设计数字电路的数学工具。逻辑变量是用来表示逻辑关系的二值量。它们取值只有0和1两种，它们代表的是逻辑状态，而不是数量大小。

逻辑代数有三种基本运算（与、或、非），应掌握逻辑代数的运算规则和基本公式。

逻辑函数通常有五种表示方式，即真值表、逻辑表达式、卡诺图、逻辑图和波形图，它们之间可以相互转换。

— End